CHAPTER

9

DevOps and Real World ROI

During the 2016 Formula 1 season, it's not unusual for pit-stops to hover somewhere around the three-second mark. In rare instances, such as for the Williams F1 team at the 2016 European Grand Prix, the feat has amazingly been accomplished in less than two ticks of the clock.

Wrap your mind around that idea for a moment. It will likely take you as long to digest the concept, and it certainly took longer to type out this sentence, than the amount of time required for a 15-20 member pit-crew to pounce on a stopping race car, lift it off the ground, swap four wheels and tires, and then drop it down to be sent on its way. Two seconds![1]

Quality Is Everyone's Responsibility

Of course, this larger performance represents a complex chain of lesser events, mapped out and practiced relentlessly by highly-paid, full-time professionals—pit-crews whose sole job is to save their teams and drivers precious seconds that could result in a change in position, and in some cases even a victory.

[1] "Williams Data Shows Baku Pit Stop a New Record," Motorsport.com, June 22, 2016: http://www.motorsport.com/f1/news/williams-data-shows-baku-pit-stop-a-new-record-790776/

A. Ravichandran et al., *DevOps for Digital Leaders*, DOI 10.1007/978-1-4842-1842-6_9

As one crew member pirouettes away with a used tire, another immediately spins into place and mounts a new wheel on the car's hub, with a pneumatic air gun operator barely removing the apparatus as one wheel is removed and another arrives in its place. This process is carried out in concert simultaneously on all four wheels of the car. Suffice it to say, to pull this act off successfully, execution must be flawless, every time.

Consider that over the 70-plus years of F1 history, at least ten individual races have been decided by less than two-tenths of a second, and many more by a matter of mere seconds; the impetus to strive toward such lofty speed and precision quickly comes into focus.[2]

Making a Case for DevOps ROI

How do these F1 crews perform such a precise ballet? By using teamwork, communication, and near constant review meant to spur process improvement. Countless hours are spent reviewing pit-stop video looking for even the smallest opportunities for potential refinement, followed by an endless cadence of hands-on practice wherein these movements are repeated hundreds of times.

Meanwhile, revised wheel, hub, and pit-crew equipment designs are introduced in the name of further speeding this process. The drive for perfection is continuously evolved as part of a cycle that never ends.

The return-on-investment (ROI) for teams that master this craft is evidenced not only in race results and unofficial pit lane bragging rights, both of which carry weight, but more importantly in hard dollars and cents. While races are often decided by seconds, the difference in prize money between the first place and tenth place finishers in the 2016 F1 Constructors Championship is estimated at roughly $150 million.[3]

Pit-stops represent only a small element of the overall competition, but when it comes down to it, every single aspect of F1 racing is aimed at shaving seconds, and even milliseconds, in the name of victory.

Inside the modern software factory, workers are engaged in a similar high-stakes race aimed at addressing an elusive opportunity, which, like a race or season win in F1, can disappear in the blink of an eye. This is crucible —succeeding in the applications economy also demands constant review, improvement, and acceleration of velocity.

---

[2]"Who Won it? 10 of F1's Closest Finishes," FOX Sports, September 1, 2014: http://www.foxsports.com/motor/story/who-won-it-10-of-f1-s-closest-finishes-090114

[3]"Formula 1 Prize Money 2016 (Constructors Championship)," Total Sportek, March 17, 2016: http://www.totalsportek.com/f1/formula-1-prize-money/

As outlined in the proceeding chapters, it's widely accepted that in order to grow and survive in the current business environment, organizations must embrace digital transformation, thereby advancing their ability to evolve software and services to meet changing customer expectations. Beyond the growing emphasis on this theme by numerous business and technology leaders, and certainly speculation on the part of technology vendors, hard data has begun to emerge which serves to cement the requirement for DevOps adoption.

For instance, in a 2015 study published by CA Technologies and Zogby Analytics, 68 percent of consumers noted that they will completely abandon a particular brand based on mere seconds of delay in application loading times.[4] Specifically, respondents indicated an expectation that applications load in six seconds or less, or else they simply drop the app, and often the related provider. Further, more than half these respondents indicated that they actually expect applications loading times of less than three seconds before they run out of patience.

Based on this reality, the need for broader adoption and advancement of DevOps culture, process, and tooling to speed and improve applications delivery is thrown into stark relief. An organization's ability to wrap its arms around DevOps has become a key differentiator in competing for end users' attention and dollars; in contrast, failing to meet changing customer expectations related to applications quality and performance often means losing out to more agile competitors.

In short, improvement of software delivery has become a prerequisite for survival. Based on that conclusion, beyond the vague notion of advancing general viability, measurement of DevOps ROI impact is clearly an increasingly critical practice.

The Challenge of Measuring DevOps Success

Measuring the impact of a process as far-reaching and transformational as DevOps within the context of ROI may seem impossible to some observers, but organizations of all kinds—from practitioners to industry analysts—have begun to attempt such calculations.

For starters, creating the necessary framework for such metrics requires closer consideration of the very reasoning behind DevOps adoption. Loosely, this rationale is accepted as the need to increase the pace of software development and release, all while improving the quality of resulting applications. Just as importantly, DevOps also brings with it the promise of speeding the rate of response to emerging performance issues, all while better addressing changing customer expectations.

---

[4] "Software: The New Battleground for Brand Loyalty," CA Technologies and Zogby Analytics, March 4, 2015: http://rewrite.ca.com/us/articles/application-economy/research-study-software-the-new-battleground.html

At first glance, this would seem a set of processes and concepts easily tied to straightforward notions of measurement, such as the rate of software deployments. However, that in itself may be too simple a viewpoint. The reality is that such measurements highlight some of the immediate benefits of DevOps advancement, while many other intrinsic results are certainly harder to quantify.

Consider that some experts, such as DevOps industry pundit and former research analyst Michael Cote, have posited that attempting to gauge the ROI of DevOps, in this dualistic sense, is "simultaneously absurd and important".[5]

The reason being, the expert maintains, that there are too many variables involved in DevOps transformation to make any overarching estimate of impact a solely believable value.

Writing for FierceDevOps in 2015, Cote specifically noted that "DevOps is an immeasurable process with respect to ROI, because its value is nearly impossible to measure independently and precisely."

That said, Cote relents that while the larger import of DevOps may be difficult to define in terms of pure ROI, it is possible and practical to measure the success of specific projects—or products developed using DevOps processes—to establish some baseline for related assessment. This is where calculations such as software deployment rates and so-called mean-time-to-repair (MTTR)—which measures the ability of organizations to identify and repair applications issues that arise in production—infer some guidelines for assessing DevOps returns, if not larger success.

As noted by industry watcher Christopher Null of *TechBeacon* in his seminal article, "How to Measure DevOps ROI,"[6], such measurement of impact is most practical within the most basic context of DevOps adoption, namely, saving any amount of time previously spent building, deploying, and maintaining applications.

"One of the biggest benefits, if not the biggest benefit, of DevOps is the promise of speed; DevOps enables more and faster code deployments, which means a decreased time to market and more opportunities to capture revenue from customers," Null observes.

Using such a framework, one can also begin to tie delivery of mission-critical, or even revenue-impacting functionality to applications lifecycle improvements garnered via DevOps processes. For these reasons, the most practical forms of DevOps ROI measurement available today do in fact revolve around factors such as software release cycles, MTTR, and applications code change

[5]"DevOps ROI," Fierce DevOps, July 21, 2015: www.fiercedevops.com/tags/devops-roi
[6]"DevOps ROI," Fierce DevOps, July 21, 2015: www.fiercedevops.com/tags/devops-roi-how-measure-guide

failure rates. These are metrics that may not conclusively outline the larger impact of the overall movement, but that can infer some notion of ongoing improvement.

While additional forms of DevOps ROI assessment are being created, including those focused on measurement of financial impact, enhanced productivity, and other indicators of organizational efficiency—for now, the most accurate metrics are those largely focused on applications lifecycle excellence and these figures do provide a compelling case for DevOps adoption and advancement.

As Null points out, Patrick Debois, himself a sysadmin, notably stated regarding examination of overriding DevOps value: "We should be thinking about [ROI] in terms of … accelerated benefits realization, and shortening that [cycle]. It's really not the ROI of DevOps, it's really more that the ROI of your original project can be realized sooner."

A Real-World Model for DevOps ROI

As widely accepted that applying specific ROI calculations within the domain of IT, in general, has always been a tricky science. This certainly applies to the domain of DevOps.

This is largely related to the fact that technology has been largely perceived as a cost-center versus a driver of profits, making it difficult to convince management to increase investment with the specific goal of improving the bottom line.

Even with scads of industry analyst calculations aimed at addressing this very issue (IT ROI), there's been little concrete evidence establishing that IT investments can in fact provide significant returns. Most experts would more readily accept that IT may become a larger channel for, or catalyst to growth of the business, but any notions or promise of massive ROI is often overshadowed by requirements for sizeable upfront and ongoing investment.

Further, technology vendors notoriously muddy the waters by proffering self-serving calculations that often inflate the proposed impact of their products in the name of boosting sales and competitive prospects. That said, it's also recognized that, especially in recent years, technology paradigms such as mobility, the cloud, and virtualization have delivered economies of scale and new business opportunities so massive as to make them difficult to fully quantify.

So, how does one go about making a believable ROI case around DevOps, a process that, like many other movements, promises to completely redefine the manner in which organizations create and monetize their value?

The 2016 State of DevOps Report, the industry's leading research project—conducted and prepared by the DevOps Research and Assessment (DORA) initiative—offers an avalanche of data providing concrete testament to the proposed, and oft-considered fuzzy, benefits of the overall movement.[7]

For starters, the report posits that so-called "high-performing organizations," those already deeply engaged in the use and maturation of DevOps practices, are "decisively outperforming their lower-performing peers in terms of throughput." This finding not only builds on similar conclusions in previous iterations of the same report, but also maintains that the espoused gap between DevOps adopters and laggards continues to widen.

Specifically, DORA contends that such high performers deploy applications code 200 times more frequently than organizations that have not yet closely embraced DevOps. So, practically speaking, such organizations have seemingly become far more capable of engaging the benefits of agile computing; namely, responding adeptly to changing customer demands.

In addition, the State of DevOps Report also finds that high-performing organizations are currently achieving 2,555 times faster lead times, serving as even greater proof of their abilities to remove obstacles to software innovation and improvement.

Perhaps even more importantly, in particular as it relates to the matter of agility and the ability to cater to evolving performance conditions, the report estimates that leading DevOps practitioners lay claim to 24 times faster recovery times and three times lower change failure rates. These figures formulate a highly compelling narrative when it's recognized that reduction of MTTR represents one of the most critical differentiators in today's applications economy.

The State of DevOps Survey, completed by roughly 4,600 technology practitioners, also found that high-performing teams spend far less time on so-called "unplanned work and rework"—most often required to respond to emerging issues related to existing applications code. This allowed them to spend nearly 50 percent more of their time on new work. The latter brand of effort is typically aimed at adding or improving feature functionality in the name of increasing business opportunities.

Lastly, in terms of pure applications delivery ROI metrics, the State of DevOps Report suggests that leading practitioners are suffering far fewer applications and services outages—those timeframes when systems are offline and truly prevent the flow of end user traffic and business.

---

[7] "2016 State of DevOps Report," DORA and sponsors, June 22, 2016: https://puppet.com/resources/white-paper/2016-state-of-devops-report

Based on a framework established by research analysts at IDC, which concludes that hourly application downtime costs can range from $1.25 to $2.5 billion for a Fortune 1000 firm, and that the average cost of a critical application failure is $500,000 to $1 million per hour,[8] the State of DevOps Report allows for the conclusion that high-performing organizations, compared even to those with some level of DevOps adoption, save an average of over $91 million per year.

Much of this advantage is found, the researchers note, in the form of lower applications change failure rates, or those updates to software code that result in subsequent performance issues or outages. While high-performing DevOps practitioners experience an average change failure rate of 7.5 percent, medium performance organizations see that figure rise dramatically, to a whopping 38 percent.

If the undeniable adage that "time is money" applies in today's applications economy as much, if not more than ever, then these ROI calculations carry substantial meaning.

Beyond such metrics aimed at measurement of improvement appreciated across the applications lifecycle, the 2016 State of DevOps Report also sheds light on another area of the movement's impact that would appear to transcend immediate calculation, but which can certainly be quantified in general terms. Organizations deeply engaged in DevOps transformation tend to have happier, more dedicated employees.

Given the wide recognition that it is far more expensive to hire and train new employees than retain existing staff, and that highly skilled IT workers in particular are difficult to find and hold on to, this less hands-on ROI benefit of DevOps may prove as even more critical support for adoption than some of the proceeding metrics.

Specifically, DORA's researchers found that employees of high-performing, DevOps-centric organizations are 2.2 times more likely to recommend their organization as a great place to work, and 1.8 times more likely to recommend their team as a great working environment. The figures were calculated using the employee Net Promoter Score (eNPS), which is designed to measure overall employee loyalty.

As a follow on, it's worth noting that other research, notably a report published by experts at Bain & Co., has found that companies with highly engaged workers typically produce revenues 2.5 times greater than those with low engagement levels. In addition, Bain submits that publicly traded stocks of companies with a high-trust work environment outperformed market indexes by a factor of three.[9]

---

[8]"DevOps and the Cost of Downtime: Fortune 1000 Best Practice Metrics Quantified," IDC, December 1, 2014: http://www.idc.com/getdoc.jsp?containerId=253155

[9]http://www.bain.com/publications/articles/the-chemistry-of-enthusiasm.aspx#2

Simply stated, these conclusions, both process-specific and more general in nature, highlight the specific effect that DevOps is having in terms of creating the most critical form of ROI measurement—the ability to innovate faster, build organizational momentum, reduce costs, and attempt to differentiate offerings within the competitive environment.

It only follows naturally that organizations spending more of their time creating new value by translating ideas into product offerings are likely outperforming those struggling to play catch-up by addressing existing shortcomings.

Measuring the Impact of DevOps Automation

As previously noted, success in every element of the F1 racing arena is measured in the smallest of increments, in this era most often in the form of tenths, or more typically hundreds of seconds. For F1 pit-crews, refinement of each tiny physical movement or adaptation of time-saving technology can result in a measurable benefit.

The same can be said of DevOps adoption, which can be extrapolated to span nearly every aspect of software design, delivery, and support, along with a multitude of tools aimed at automating some involved process.

To isolate any one or two specific processes or tools only provides a small window into the broader impact of DevOps automation as a whole; however, leveraging these as a starting point can provide perspective which can then be expanded across numerous domains.

One such toolset that has been so examined is the utilization of CA Agile Requirements Designer solution, which seeks to automate test case design and execution based on changing requirements—to automate one of the most time-consuming and resource-intensive problems in applications delivery—namely, delays created by manual testing.

By allowing engineering teams to automate the process of creating and applying software testing requirements, the solution promises to greatly reduce related cost and complexity, specifically by directly cutting down the number of required test cycles.

According to a calculation designed to forecast the potential ROI of CA Agile Requirements Designer related to cost reduction and avoidance, along with related revenue enhancement, immediate benefits of employing this process—representing only one minor aspect of DevOps adoption—immediately come into focus.

Based on a real-world implementation of the solution, in addition to gains related only to software testing, one organization found that it was able to drive far closer collaboration between business and IT teams, and better

estimate the time and cost of development projects. In addition to the following ROI calculations, these gains map closely to core benefits of broader DevOps adoption.[10]

In terms of specific ROI, the involved organization found that its worst case appreciation of benefits over a three-year period would result in a 168 percent return on its existing investment, or a period of 11 months before 100 percent realization of payback on initial costs; the study projected a best case scenario of 319 percent ROI, or 8 months for complete payoff.

Among the underlying benefits driving this return were the organization's ability to reduce software defect remediation (a similar domain to reduction of MTTR), lowering of test creation and maintenance costs, and, most importantly within the overall scope of strategic investment, improved time to value for application releases.

At the production end of the software lifecycle, another important DevOps tooling aspect that warrants ROI analysis is Application Performance Management (APM). APM seeks to simplify the triage of performance problems so teams can address issues fast, before there's an impact on end users and customers.

By quickly analyzing performance across a complex array of microservices, APIs and containers across mobile, cloud, and on-premise applications, these monitoring tools provide IT operations an essential mechanism for maintaining expected levels of service together with a high-quality application experience.

In a DevOps context and as described in Chapter 7, these tools are especially valuable because they help establish feedback loops between operations and development teams. By gaining insights into application performance before committing software updates to production, developers can quickly pinpoint and remediate any issues related to their code. In this way, APM solutions not only help ensure operational stability, but also assist teams in building quality into the applications.

According to a 2015 Total Economic Impact study conducted by Forrester Research, the potential ROI of CA APM is related to significantly reducing downtime costs and improving productivity.[11]

---

[10]"CA Test Case Optimizer ROI Business Case Detailed Report," CA Technologies, July 1, 2015: https://km.ca.com/sales/presales/roi/ROI%20Portal/Dashboard/CA%20Test%20Case%20Optimizer%20Reference%20ROI%20Business%20Case%20V1FY16REF.pdf

[11]The Total Economic Impact of Application Performance Management: http://www.ca.com/content/dam/ca/us/files/industry-analyst-report/the-total-economic-impact-of-apm.pdf

Interviews with four customers (across financial services, banking, and health insurance sectors) and subsequent financial analysis revealed that a composite organization based on the interviewed organizations experienced benefits of more than $7.2 million versus implementation costs of just over $1.7 million, adding up to a net present value (NPV) of $5.4 million and an ROI of 306 percent.

As would be expected, all the interviewed organizations experienced a positive financial impact by improving the availability and performance of customer-facing applications. For one company, this extended to reducing service-level penalty fees because it was able to prove the cause of problems to a key commercial customer, reducing the flow of more than $800,000 that it was paying yearly to a large, global vendor.

Beyond availability gains, and importantly from a DevOps perspective, there were other benefits. The study found that by applying CA APM within the development process, developer productivity increased. When the development team used CA APM to test new code, the organization was able to accelerate development cycles by 15 percent in the first year. As a representative of one company said, "In essence, we're issuing less code or less fixes into production, which allows our developmental organization to focus more on maintaining a competitive advantage in the marketplace of what we deliver in our online solutions."

Leveraging CA APM provided insight into performance from mainframes through mobile devices. Using CA APM allowed the organization to eliminate uncertainty about the cause of problems in complex environments. CA APM also reduced the average time to resolution and it eliminated the finger-pointing.

Avoiding blame games and a hero culture is of course the essence of DevOps, but application complexity and needing to support multiple stakeholders means tools must be capable of delivering uninterrupted visibility. The study illustrated that by providing performance insight from mobile to mainframe, CA APM allowed the organization to eliminate the uncertainty about the absolute cause of problems in complex environments. As one performance analyst from a regional consumer bank stated, "APM gives us micro-visibility into the customer experience."

The Whole Is Greater than the Sum of its Parts

Given that these aspects (agile requirements design and APM) of applications delivery represent only parts of the overall lifecycle and areas for potential DevOps impact, one can easily justify adoption based on the ROI they singularly deliver. However, to be truly reflective of DevOps impact, tools should be examined in unison. This way, organizations can begin to extrapolate how combinations across the automated software factory can create the massive gains highlighted in the State of DevOps Report.

Figure 9-1 and the notes that follow illustrate a simple but powerful example of the cumulative benefit of combining automated tools across the software lifecycle.

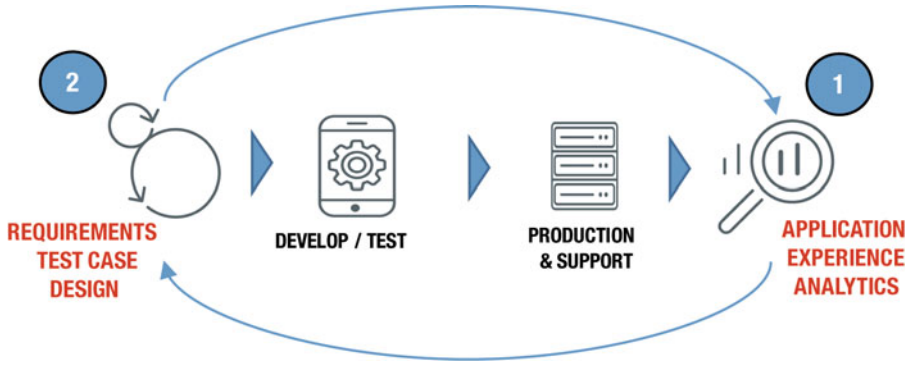

Figure 9-1. Cumulative benefits of combining multiple tools example: application experience analytics and agile requirements design

Step 1: Application Experience Analytics

In this step, tools feed information back to agile teams, which is used as input for requirements changes and iterative development.

Expected benefits include constant cycling of intelligence gained through analytics (e.g., app and functional usage and live performance) that helps developers, operations, and the business improve applications with every release.

Step 2: Requirements and Test Case Design

As requirements change, agile requirements design tools automate test case design and their execution.

Expected benefits include maximum functional coverage with the smallest number of tests, as well as reduction in test case cycles by automatically removing any outdated, redundant, or duplicate test cases.

When other tools are integrated, the benefits will continue to accumulate. For example, by automatically providing synthetic data from a test data warehouse and matching this directly to the test cases created in Step 2, an organization can:

- *Improve quality*—Providing the right data for testing can reduce defect creation
- *Improve testing efficiency*—Eliminating data constraints can reduce time and resources needed to provision data

- *Reduce costs*—Creating accurate subsets of data can reduce infrastructure cost

Summary

As you've seen, current measurement of DevOps ROI remains somewhat nascent, and difficult. However, real-world use cases and industry research clearly highlight the massive impact that the adoption of DevOps practices is having among today's early adopters. The future looks bright for those who get on board, while those left behind may struggle to compete and survive.

In the next and final chapter, we'll look at some additional concepts, techniques, and emerging trends that have begun to populate the DevOps landscape and examine how these may be used to enhance a DevOps program.